ER2FGSM: Efficient and Robust Adversarial Attacks Against Deep Reinforcement Learning

Abstract

7 pages + 2 pg ref.; full paper deadline: 9/8

The high performance of Deep Reinforcement Learning (DRL) algorithms in complex tasks has increased the need to ensure the robustness of the DRL against advanced adversarial attacks. However, compared to a large volume of studies exploring adversarial attacks that perturb Deep Neural Networks (DNN), adversarial attacks to disrupt DRL have been significantly less explored. The existing adversarial algorithms for DRL have focused on attacking a DRL agent mainly disrupting its decision making. However, little work has explored in injecting perturbation attacks in input data (i.e., adversarial examples) in the context of DRL settings. In this paper, we propose efficient and robust perturbation-based adversarial attacks to disturb the DRL agent's decision making, called *Efficient and Robust Randomized Fast Gradient Sign Method* (ER2FGSM). We significantly refined the Randomized Fast Gradient Sign Method (RFGSM) previously studied under DNN settings to be applicable under DRL environments. We enhanced it further by incorporating momentum on each gradient update. We considered targeted or non-targeted attacks under DRL with or without defenses to investigate the efficiency, effectiveness, and robustness of the ER2FGSM. We conducted extensive experiments using Deep Q-Network (DQN) agent (i.e., DRL agent) playing an Atari Pong game where our ER2FGSM is compared against the five state-of-the-art DRL attacks in terms of attack success rate, attack execution time, and robustness under defenses. Our results proved that our proposed ER2FGSM attack is 17 times faster than the state-of-the-art Carlini & Wagner (CW) method while showing highly comparable attack success rate. Our results also proved that ER2FGSM is more robust against the state-of-the-art defense than its existing counterparts.

Introduction

Deep Reinforcement Learning (DRL) algorithms learn policies to guide DRL agents to take optimal actions based on an environment state. These algorithms have successfully achieved high performance on the various complex as well as critical tasks, such as robotics (Amarjyoti 2017), autonomous vehicles (Kiran et al. 2021; Ferdowsi et al. 2018), resource allocation (Yoon et al. 2021b), intrusion response systems (Yoon et al. 2021a), various cybersecurity problems (Basori and Malebary 2020), or networking and communication problems (Luong et al. 2019). As DRL has been used to solve multiple problems as above, adversaries aiming to disrupt the DRL process and misleading the DRL agent's decision-making have been known as a severe issue.

A policy, a probabilistic distribution of actions by the DRL agent, is often learned by Deep Neural Networks (DNN) to approximate the action-value function. The vulnerabilities of the DNN to adversarial attacks have been significantly studied (Szegedy et al. 2013; Goodfellow, Shlens, and Szegedy 2014a; Yuan et al. 2019) to mitigate the impact when they are exploited by adversaries. Common adversarial examples include adversarial perturbations imperceptible to humans but fooling DNNs easily in the testing or deploying stage (Yuan et al. 2019). Researchers have expired various attacks and defenses for supervised DNN applications, such as image classification (Goodfellow, Shlens, and Szegedy 2014a) or natural language processing (Alzantot, Balaji, and Srivastava 2018). However, adversarial attacks and defenses are largely unexplored in DRL environments. DRL has numerous critical safety and security applications and accordingly drew our attention to the need for robust DRL. For robust DRL, there is a prerequisite of developing efficient, effective, and robust adversarial attacks which can evaluate the robustness of defense mechanisms.

Researchers have developed adversarial attacks in DRL by answering the following two questions: (1) *How to attack?*; and (2) *When to attack?*. The first *how-to-attack* question is related to what perturbation method to use for disrupting the state during an episode. The second *when-to-attack* question is related to identify an optimal time to attack during an episode. In this work, we focus on answering *how-to-attack* by proposing the so-called *Efficient and Robust Randomized Fast Gradient Sign Method*, namely ER2FGSM. To be specific, **the goal of this work** is to develop robust and fast attacks by generating effective adversarial states in DRL. In addition, we validate the performance of the proposed ER2FGSM by comparing it against those of the state-of-the-art adversarial attacks under DRL with or without defense where the attacker can perform targeted or non-targeted attacks. We validated the performance of the ER2FGSM in terms of attack success rate (ASR), average attack execution time per perturbation (AET), and av-

erage reward (AR) by the DRL agent via the extensive comparative performance analyses.

We made the following **key contributions** in this work:

1. We developed the ER2FGSM, which generates fast and robust adversarial perturbations to compromise a DRL agent under either targeted or non-targeted perturbation attack. We considered Deep Q-Network (DQN) as a DRL algorithm playing an Atari Pong game, enabling defense against any perturbation attacks. ER2FGSM is an enhanced version of RFGSM (Tramèr et al. 2017), Momentum Iterative FGSM (MI-FGSM) (Dong et al. 2018), and Iterative FGSM (I-FGSM) (Kurakin et al. 2016), which can efficiently attack the states of the DRL process. In our ER2FGSM, instead of identifying an optimal time to attack in the DRL process, we considered attacking all steps during an episode. This approach allowed evaluating the effectiveness (i.e., ASR) of the EF2FGSM on all situations during the episode. In addition, this enabled a fair comparison with other existing perturbation attacks.
2. We considered Robust ADversarIAL Loss DQN (RADIAL-DQN) (Oikarinen, Weng, and Daniel 2020) as a more robust defense in the DQN and investigated its defense effect in DRL on the robustness of the ER2FGSM attack in ASR, compared to those of the five state-of-the-art attacks.
3. To validate the outperformance of the proposed ER2FGSM, we conducted extensive performance analysis to compare the ER2FGSM with the five state-of-the-art adversarial examples in DRL, including (Carlini and Wagner 2017; Goodfellow, Shlens, and Szegedy 2014b; Madry et al. 2017; Dong et al. 2018; Xie et al. 2019), in terms of ASR, AET, and AR. Among the five existing adversarial examples, we devised two baseline examples, which are more robust and scalable, by extending the Diversity Iterative FGSM (DI-FGSM) (Xie et al. 2019) and Momentum Iterative FGSM (MI-FGSM) (Dong et al. 2018) used in DNNs into a DRL setting. Our results showed that our proposed ER2FGSM significantly outperformed the state-of-the-art Carlini & Wagner method (CW) in AET while maintaining ASR and AR comparable to other counterparts. In terms of ASR and AR under the defense, ER2FGSM outperformed the state-of-the-art perturbation methods, proving its robustness.

Related Work

This section provides a brief overview of the state-of-the-art adversarial attacks in Deep Neural Networks (DNN), mainly used for supervised deep learning. In addition, we briefly discuss existing adversarial attacks to disrupt the DRL process and identify the differences between our proposed ER2FGSM and their approaches.

Backdoor attacks in Deep Neural Networks (DNNs). Szegedy et al. (2013) initially presented an attack with small perturbations leading to misclassifications. Later, researchers have proposed various attacks and defenses and evaluated them in terms of scalability, attack success rate (ASR), and robustness. Goodfellow, Shlens, and Szegedy (2014a) proposed the Fast Gradient Sign Method (FGSM), which was efficient but turned out to be less robust, not guaranteeing a 100% success rate. Carlini and Wagner (2017) proposed the Carlini & Wagner (CW) method that guarantees 100% ASR but was slow. Naïve FGSM provided the basis to build more sophisticated and better variants of FGSM, such as Randomized Fast Gradient Sign Method (RFGSM) (Tramèr et al. 2017), Diversity Iterative Fast Gradient Sign Method (DI-FGSM) (Xie et al. 2019), and Momentum Iterative Fast Gradient Sign Method (MI-FGSM) (Dong et al. 2018). Madry et al. (2017) proposed the Projected Gradient Descent (PGD), a variant of Iterative FGSM using projected gradient descent, making it robust. Currently, MI-FGSM and PGD attacks are considered the most efficient and robust state-of-the-art adversarial examples in DNNs. In particular, RFGSM, DI-FGSM, and MI-FGSM are only designed and evaluated in the context of DNNs and have never been used in DRL. In this work, we proposed ER2FGSM by enhancing RFGSM and incorporating momentum and applied it in DRL. In addition, we also refined DI-FGSM and MI-FGSM to be applied in DRL and be compared against that of the ER2FGSM.

Adversarial Attacks in Deep Reinforcement Learning (DRL). Huang et al. (2017) made the preliminary attempt to attack the DRL agent by extending FGSM. Later, state-of-the-art attacks of DRL, such as Lin et al. (2017); Sun et al. (2020), majorly focused on finding the optimal time to attack the DRL agent. However, Lin et al. (2017); Sun et al. (2020) ignored the question of *how-to-attack* and blindly used the state-of-the-art state perturbation technique, such as Carlini & Wagner (CW) method. However, we demonstrated that CW is very slow and less robust. Therefore, we just compared our attack with the CW-based perturbation attacks to prove the effectiveness of ER2FGSM.

Fast perturbation methods, such as naïve Fast Gradient Sign Method (FGSM) (Goodfellow, Shlens, and Szegedy 2014b), have been used in DRL by (Huang et al. 2017) ; however, the naïve FGSM is easily detectable. Another variant of FGSM, called PGD, is one of the state-of-the-art DRL attacks as it is fast and more robust than the CW method. However, the state-of-the-art defenses in DRL (Zhang et al. 2020; Oikarinen, Weng, and Daniel 2020; Fischer et al. 2019) have recently challenged the robustness of PGD-based attacks (Madry et al. 2017), which were originally considered in DNNs. Therefore, there is a critical need to develop scalable, effective, and robust state perturbation attacks in DRL settings, which is the goal of our paper.

Preliminaries

This section provides an overview of the DRL and adversarial states generation methods considered in this work.

Deep Reinforcement Learning (DRL)

Reinforcement learning (RL) algorithms optimize the expected cumulative reward by training a policy π. This policy can be a deterministic or probabilistic function that maps state s to action a, $\pi : S \rightarrow A$, where S and A are state and action spaces, respectively. In DRL, this policy function π is learned by a neural network. Deep Q-Networks

(DQN) (Mnih et al. 2013) is one of the well-known DRL algorithms. We have evaluated our attacks on the policy trained by DQN.

Deep Q-Networks (DQN). In Q-learning, the Q-value is the expected cumulative discounted reward when action a is taken in state s. DQN is a kind of Q-learning where Q-values are learned by a neural network (NN) while minimizing the squared Bellman error. DQN has the component, called *Experience Replay*, which helps decrease the high variance due to Q-learning updates. Another component, called *Replay Buffer*, stores all recent transitions. To mitigate the correlation due to time, random samples are taken from this buffer. DQN takes action based on the maximum Q-value.

Adversarial States Generation Methods

We will compare the performance of the following methods (or variants of them) against that of our ER2FGSM.

Fast Gradient Sign Method (FGSM). FGSM (Goodfellow, Shlens, and Szegedy 2014a) focuses on attack efficiency to design adversarial examples, rather than optimal performance of adversarial examples. For an image x, a perturbed image x' according to FGSM is:

$$x'_{\text{FGSM}} = x + \epsilon \cdot \text{sign}(\nabla \text{loss}(h(x), y_{\text{true}})), \quad (1)$$

where t is a target label and ϵ is a parameter to make the perturbation small enough to be less noticeable. The loss function can be a cross-entropy function. FGSM uses the linear approximation of the model and solves the maximization problem in a closed form, which makes this method very fast. There are three kinds of norm constraints, L_0, L_1, and L_∞, which are used in the literature to constrain the perturbation to be undetectable. For n to be zero, one or infinite norm, the constraint can be presented by:

$$||x - x'||_n < \epsilon. \quad (2)$$

Carlini & Wagner Method (CW). Carlini and Wagner (2017) presented three attacks guaranteeing 100% ASR; however, they are much slower than naïve FGSM (Goodfellow, Shlens, and Szegedy 2014b). CW optimizes an objective function in a multi-step process to generate minimal and optimal perturbations, making the perturbation generation very slow.

Projected Gradient Descent (PGD). Madry et al. (2017) proposed PGD, which is a multi-step variant of FGSM, utilizing the projected gradient descent on the negative loss function. This is also the attempt to solve the inner maximization problem. Starting with $x'_0 = 0$, on every iteration, it performs:

$$x'_i = \prod_{x+\delta} (x'_{i-1} + (\alpha \cdot \text{sign}(\nabla \text{loss}_{F,t}(x'_{i-1})))). \quad (3)$$

Diversity Iterative FGSM (DI-FGSM). In I-FGSM, there is an overfitting problem if the number of steps/iterations increases. Xie et al. (2019) proposed DIFGSM to solve the overfitting problem in I-FGSM. At each iteration, unlike the traditional methods which maximize the loss function directly w.r.t. the original inputs, DIFGSM applies random and differentiable transformations (e.g., random resizing, random padding) to the input images with probability p and maximizes the loss function w.r.t. these transformed inputs. This method is also similar to ER2FGSM as they both try to generate hard and diverse input patterns. This was primarily developed for blackbox settings to preserve generalizability and diversity. This is also similar to I-FGSM with the addition of transformation T. Starting with $x'_0 = 0$, on every iteration, it performs:

$$x'_i = x'_{i-1} - \text{clip}_\epsilon(\alpha \cdot \text{sign}(\nabla \text{loss}_{F,t}(T(x'_{i-1})))). \quad (4)$$

We leveraged the following three methods (i.e., I-FGSM, MI-FGSM and RFGSM) to develop our ER2FGSM.

Iterative Fast Gradient Sign Method (I-FGSM). Kurakin et al. (2016) proposed I-FGSM, a variant of FGSM, taking multiple small steps of size α in the direction of the sign of gradient. However, FGSM takes only one step of size ϵ to make perturbation small enough. This method also clips the result of multiple steps by ϵ. I-FGSM outperformed FGSM. Starting with $x'_0 = 0$, on every iteration, it performs:

$$x'_i = x'_{i-1} - \text{clip}_\epsilon(\alpha \cdot \text{sign}(\nabla \text{loss}_{F,t}(x'_{i-1}))), \quad (5)$$

where α is a step in the direction of the sign of gradient.

Randomized FGSM (RFGSM). A single-step FGSM method has a problem of converging to degenerate a global minimum. I-FGSM obfuscates a linear approximation of the loss due to small steps. Due to these problems, they generate weak perturbations, which can be easily defended. To tackle this problem, Tramèr et al. (2017) presented RFGSM, which adds a small random step to such single-step attacks to escape the non-smooth vicinity of the data point before linearizing the model's loss. Single-step RFGSM is computationally efficient and has a high ASR than I-FGSM in DNNs. It performs:

$$x^{\text{adv}}_{RFGSM} = x' + (\epsilon - \alpha) \cdot \text{sign}(\nabla_{x'} \text{loss}(x', y_{\text{true}})), \quad (6)$$

where

$$x' = x + \alpha \cdot \text{sign}(\mathcal{N}(0^d, I^d)). \quad (7)$$

Momentum Iterative FGSM (MI-FGSM). Dong et al. (2018) proposed MI-FFSM, which is a variant of FGSM that integrates the momentum on each step of an iterative FGSM to escape from poor local maxima and stabilize update directions. Starting with $x'_0 = 0$, on every iteration, it performs:

$$g_i = \mu \cdot g_{i-1} + \frac{\nabla \text{loss}_{F,t}(x'_{i-1})}{||\nabla \text{loss}_{F,t}(x'_{i-1})||_1}, \quad (8)$$

$$x'_i = x'_{i-1} - \text{clip}_\epsilon(\alpha \cdot \text{sign}(g_i)), \quad (9)$$

where μ is the decay factor and g_i is the accumulated gradient at iteration i.

Problem Statement

A DRL agent interacts with an environment and learns policy π to choose action a given a state s. This policy π can be a probabilistic model $\pi(\text{s}, \text{a}) \sim [0, 1]$, which gives the probability of taking action a given a state s. The π can also be a deterministic model where we can obtain action a directly

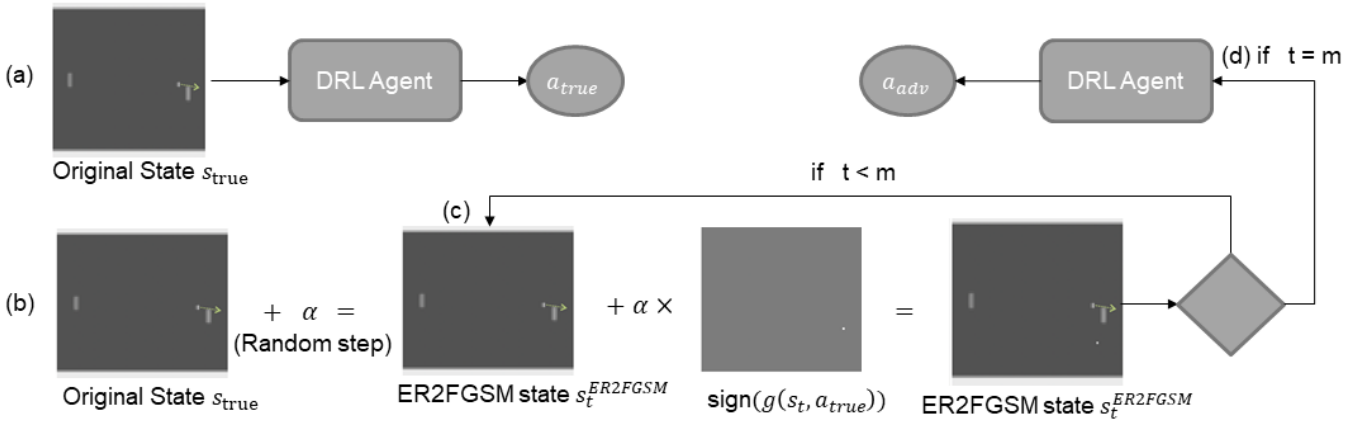

Figure 1: The overview of Non-Targeted ER2FGSM: ER2FGSM attacks a single frame during an episode to compromise a DRL agent following: (a) DRL agent takes an true non-adversarial state s_{true} to give non-adversarial, true action, a_{true}; (b) Start by adding the random step of size α to s_{true}; (c) Until the number of steps, m, compute the ER2FGSM state by calibrating the momentum-based accumulated gradient, g, using s_{true} and a_{true}, and then clipping it with α; and (d) Compute adversarial action, a_{adv}, by giving a final ER2FGSM adversarial state to the DRL agent. can you also use mathrm for a_{true} in the figure?

from policy function π given the state $s : a = \pi(\text{s})$. The goal of the DRL agent is to maximize the cumulative reward R_o by learning a policy π^*. The reward that the DRL agent aims to maximize is the expected discounted reward until the next $T-1$ time steps, represented by:

$$R_o = \sum_{t=0}^{T-1} E_{a_t \sim u(s_t)} \Big[\gamma^t r(s_t, a_t) \Big]. \tag{10}$$

Instead of maximizing this reward, an adversary aims to minimize reward R_o by adding perturbation δ_t into the agent's observation s_t to mislead the agent to take adverse action a_t. The adversary has to generate perturbation δ as small as possible to be undetected. Out of all time steps, the adversary may or may not choose to add perturbation δ to the state s_t based on his strategy. In this way, the adversary's expected cumulative reward is given by:

$$R_{\text{adv}} = \sum_{t=0}^{T-1} E_{a_t \sim u(s_t + x_t \delta_t)} \Big[\gamma^t r(s_t, a_t) \Big], \tag{11}$$

where x_t at given time t is 0 if the adversary does not inject any perturbation; otherwise, it returns 1.

We need to find perturbations $\delta_0, \delta_1, \ldots, \delta_{T-1}$ for all time steps during an episode to compromise the DRL agent to take adversarial actions $a_0^{\text{adv}}, a_2^{\text{adv}}, \ldots, a_{T-1}^{\text{adv}}$. These adversarial actions should then minimize the reward of the DRL agent, R_o, and the reward of the DRL agent under defense, R_{defense}. We also want to find each perturbation δ_t in the minimum time possible.

Proposed Approach: ER2FGSM

We propose the Efficient and Robust Randomized Fast Gradient Sign Method (ER2FGSM) to find robust and effective perturbations in the minimum time possible. ER2FGSM is designed by combining RFGSM, MI-FGSM, and I-FGSM and extending their applications from DNNs to DRL.

For the extension to the DRL setting, we consider true label y_{true} as a vector of weights for each action produced by the policy of DRL agent, represented by:

$$a_{\text{true}} = \text{DRLAgent}(s_{\text{true}}), \tag{12}$$

where s_{true} is the original non-adversarial state and a_{true} is the original non-adversarial action given by the DRL agent.

In ER2FGSM, we start with taking a random step of size α as in Eq. (13). Then, at each step t, we calculate the adversarial state $s_t^{ER2FGSM}$ by taking the step of size α in the direction of gradient g. Starting with:

$$s_0^{ER2FGSM} = s_{\text{true}} + \alpha \cdot \text{sign}(\mathcal{N}(0^d, I^d)). \tag{13}$$

On every iteration i, it performs:

$$g_t = \mu \cdot g_{t-1} + \frac{\nabla_{s'} \text{loss}(s', a_{\text{true}})}{||\nabla_{s'} \text{loss}(s', a_{\text{true}})||_1}, \tag{14}$$

$$s_t^{ER2FGSM} = s_{t-1}^{ER2FGSM} + \alpha \cdot \text{sign}(g_t), \tag{15}$$

where μ is the decay factor and g_t is the accumulated gradient at iteration t.

$$a_{\text{adv}} = \text{DRLAgent}(s^{\text{ER2FGSM}}). \tag{16}$$

Eq. (15) means that we move policy π away from an optimal action by using the direction of the gradient. We make it iterative to take multiple gradient direction steps. Multiple gradient steps move the policy away from the optimal action, contributing to generating good ASR. Adding a random step in the start contributes to enhancing this method robust as it allows to escape the non-smooth vicinity of the data point before linearizing the model's loss. Attack effectiveness on even a fewer gradient steps and computationally inexpensive calculation of gradients make ER2FGSM fast enough. Eq. (14) shows the addition of a momentum term in gradient on each step. This solves stucking at poor local maxima and gives the gradient a necessary boost, called a momentum,

to try to reach the global or optimal maxima. The momentum also stabilizes the gradient updates. We summarize the procedures of ER2FGSM in Figure 1.

As discussed earlier, our proposed ER2FGSM attacks all steps during an episode, allowing the evaluation of the effectiveness, efficiency, and robustness of ER2FGSM in all situations during the episode. In addition, this enables a fair comparison of our ER2FGSM with the other state-of-the-art attacks. We always consider $x_t = 1$ in Eq. (11) as we attacked all number of steps during the episode by $\sum_{t=0}^{T-1} x_t = \text{N}$ where N is the same as the length of an episode.

Experimental Setup

We use the following **metrics** for our performance analyses:

- **Attack Success Rate (ASR)** measures the total number attack success over the total number of attack attempts. Considering either targeted or non-targeted attacks, ASR is measured by:

$$\text{ASR}_{NT} = \frac{N_{NT}^{AS}}{N_{NT}}, \;\; \text{ASR}_{T} = \frac{N_{T}^{AS}}{N_{T}}, \tag{17}$$

  where N_{NT}^{AS} refers to the total number of successes by non-targeted attacks and N_{NT} is the total number of attempts by non-targeted attacks. Similarly, N_{T}^{AS} refers to the total number of successes by targeted attacks and N_T is the total number of attempts by targeted attacks. Under non-targeted attacks, a failure indicates the case where the attack is entirely unable to succeed or the DRL agent takes an action maximizing the reward. An attack attempt is defined by one time attack per time step. On the other hand, targeted attacks mean generating a perturbation targeting to lead the DRL agent to take a targeted action. For example, in Atari Pong, a targeted attack can aim to make the DRL agent take a targeted action 'up' at a given point. Thus, reward reduction by other causes, rather than the perturbation targeted attack, is not simply considered as attack success in ASR_T.
- **Average Attack Execution Time Per Perturbation (AET)** captures the average time required to generate a perturbed state. For the respective attacks, targeted or non-targeted, we measure AET by:

$$\text{AET}_{T} = \frac{\mathcal{T}(N_{NTS})}{N_S}, \;\; \text{AET}_{T} = \frac{\mathcal{T}(N_{TS})}{N_S}, \tag{18}$$

  where N_S is the total number of adversarial states, $\mathcal{T}(N_{NTS})$ is the total times elapsed to generate all non-targeted attacks, and $\mathcal{T}(N_{TS})$ is the total times elpased to generate all targeted attacks.
- **Average Reward (AR)** measures the average reward of the rewards of all episodes. Given N_e be the number of episodes and r_i be the total rewards accumulated during an episode, AR is measured by:

$$\text{AR} = \frac{\sum_{i=0}^{N} r_i}{N_e}. \tag{19}$$

Comparing Schemes

We use the following perturbation attacks to be compared against our ER2FGSM. As discussed earlier, we did not consider Lin et al. (2017); Sun et al. (2020) as comparing schemes because they focus on answering the second question, *when-to-attack*, rather than *how-to-attack*. The comparing schemes include:

- **Fast Gradient Sign Method (FGSM)** (Goodfellow, Shlens, and Szegedy 2014b): We chose this as our baseline because it has been not only extensively used in DNN settings but also has been numerous times used in DRL settings, such as *uniform attack* (Huang et al. 2017).
- **Carlini & Wagner Method (CW)** (Carlini and Wagner 2017): This method is considered one of the well-known state-of-the-art methods in DRL settings guaranteeing 100% ASR. The state-of-the-art end goal-based DRL attacks (Lin et al. 2017; Sun et al. 2020) use the CW method to generate targeted perturbations.
- **Projected Gradient Method (PGD)** (Madry et al. 2017): PGD is not much popular in end goal-based DRL attacks, but it is considered to be the robust, fast, and successful state-of-the-art attack in DRL settings. Due to its robustness, PGD is usually employed by defenders to test their defenses as in (Zhang et al. 2020; Oikarinen, Weng, and Daniel 2020; Fischer et al. 2019).
- **Momentum Iterative Fast Gradient Sign Method (MI-FGSM)** (Dong et al. 2018): We extended MI-FGSM from DNN settings to DRL settings and compared it with our method. MI-FGSM is the state-of-the-art adversarial example in DNN settings. It is considered to be the most robust and effective yet fast technique. This effective method has not been enhanced to be applied in DRL before.
- **Diversity Iterative Fast Gradient Sign Method (DI-FGSM)** (Xie et al. 2019): We also extended DI-FGSM from DNNs to DRL and compared it with our method. We included it as our baseline due to its similarity with ER2FGSM in considering randomization.

Tuned Parameters

We tuned the following parameters for the optimal setting:

- *Size of a perturbation* (ϵ): It should not be very small or very large, so being optimal to be effective but undetected.
- *Number of steps of perturbation* (m): We need to make the steps as low as possible to reduce the attack execution time while achieving high ASR.
- *Size of the step* (α): It defines the size of a step in a perturbation method taking in iterative methods with $\alpha < \epsilon$.

We have extensively parametrized each attack to identify the optimal parameter values for comparison. We summarized them in Table 1.

Defense Setting in DRL

There are very few defenses proposed for adversarial attacks in the DRL. The conventional adversarial training defense has been widely used in DNNSs (Kurakin, Goodfellow, and Bengio 2016; Madry et al. 2017) and DRL (Kos and Song 2017; Pattanaik et al. 2017; Behzadan and Munir

Perturbation Method	Number of Steps	ϵ	α
CW	1000	NA	NA
PGD	40	8/255	2/255
DI-FGSM	20	8/255	2/255
MI-FGSM	5	8/255	2/255
FGSM	1	2/255	NA
ER2FGSM	8	16/255	8/255

Table 1: Optimal values of the parameters identified for each perturbation method considered in this work

2017) as well. However, recently more robust and efficient defense methods were proposed specifically designed for DRL, such as Robust ADversarIAl Loss DQN (RADIAL-DQN) (Oikarinen, Weng, and Daniel 2020) and State Adversarial DQN (SA-DQN) (Zhang et al. 2020). We use RADIAL-DQN to test the robustness of our attacks against their attack detectability because it outperformed all other defenses in DRL (Oikarinen, Weng, and Daniel 2020).

Robust ADversarIAl Loss DQN (RADIAL-DQN). RADIAL-DQN (Oikarinen, Weng, and Daniel 2020) improved the robustness of DRL agents by designing the adversarial loss functions with robustness verification bounds during training. It leverages robustness verification bounds to keep the loss as low as possible because low loss leads to better performance of the DRL agent. It primarily parametrizes k to minimize the loss where $L_{adv} = kL_S + (1-k)L_W$. RADIAL-DQN applies robustness verification algorithms on the DQN to obtain the layer-wise output bounds of Q-Networks and uses these output bounds to calculate an upper bound of the original loss function under worst-case adversarial perturbation L_W, given L_S is the standard loss function.

Environmental Setup

We consider a white-box attack where an adversary does not have access to the training time. However, the adversary has the test time access of the DRL agent where it knows the network architecture and has access to craft its adversarial state. For our experiments, we trained Deep Q-Network (DQN) using the implementation of (Mnih et al. 2015) to play an Atari Pong game using Pytorch and Open AI Gym. The loss function is the cross-entropy loss for gradient-based attacks. We have used L_∞ norm in all FGSM based baselines, including ER2FGSM. For the CW, we have used L_2 norm. Assuming that the attacker will have the computational power equivalent to the commodity CPUs, we used Google Colaboratory CPU (AMD EPYC 7B12, 2 CPUs @ 2.3 GHz, 13 GB RAM) for our experiments.

We run every experiment 100 times to minimize the changes in metrics due to random seeds and environmental factors. From the results obtained from this experiment, we reported the average and standard deviation of AR and AETs for each experiment.

Experiments & Results

We conducted extensive experiments to evaluate the efficiency, effectiveness, and robustness of ER2FGSM under the following scenarios: (1) Non-targeted attack under no defense in DRL; (2) Targeted attack under no defense in DRL; (3) Non-targeted attack under defense in DRL; and (4) Targeted attack under defense in DRL.

In non-targeted attacks, we select an action that can minimize the reward as much as possible. On the other hand, we define the targeted attack as a special perturbation that can mislead a DRL agent to the desired action. To test our perturbation methods in a DRL setting, we gave random actions to perturbation methods to evaluate their ability to generate a targeted action. We provided the details of the designed algorithm in Algorithm 1.

Algorithm 1: **Evaluation Experiment**

Inputs:
$A \leftarrow$ an actions set)
$s_0 \leftarrow$ an initial non-adversarial state
Parameters:
targeted $\leftarrow$ return 1 if attack is targeted; 0 otherwise
defense $\leftarrow$ return 1 if defense is applied; 0 otherwise
PerturbationMethod() $\leftarrow$ a perturbation method used
for each episode **do**
  for each step during an episode **do**
    if targeted is true **then**
      $a_t^{adv*} = \text{RandomStrategy}(A)$
      $s_t^{adv} = \text{PerturbationMethod}(s_t, a_t^{adv*})$
    else
      $s_t^{adv} = \text{PerturbationMethod}(s_t)$
    end if
    if defense is true **then**
      $a_t^{adv} = \text{DQN}_{\text{defense}}(s_t^{adv})$
    else
      $a_t^{adv} = \text{DQN}(s_t^{adv})$
    end if
    $r_t^{adv}, s_{t+1}, \text{done} = \text{Perform}(a_t^{adv})$
    if done is true **then**
      break
    end if
  end for
end for

edited by here

Performance Analysis in ASR.
Needs to be done after complete results, Table 3 and histogram

Performance Analysis in AR.
Needs to be done after complete results, Table 2 and histogram

Performance Analysis in AET.
Needs to be done after complete results, Table 4 and histogram

Performance Analysis Under the Same Number of Steps. To do the fair comparison, we are fixing the number of steps to 8 as ER2FGSM gives optimal performance on 8 number of steps. Needs to be done after results, three histograms maybe or the table?

Sensitivity Analysis of ER2FGSM.

Perturbation Method	No Defense		Defense with RADIAL-DQN	
	Non-targeted	Targeted	Non-targeted	Targeted
CW (1000 steps)	-21.00 ± 0.00		$+20.85 \pm 0.36$	$+20.50 \pm 0.50$
CW (20 steps)	-21.00 ± 0.00	$+20.75 \pm 0.43$	$+20.70 \pm 0.46$	$+20.80 \pm 0.40$
PGD (20 steps)	-21.00 ± 0.00	-20.39 ± 0.8	-20.96 ± 0.20	-20.44 ± 0.80
DI-FGSM (20 steps)	-21.00 ± 0.00	-19.97 ± 1.27	-19.87 ± 1.32	-16.78 ± 2.67
MI-FGSM (20 steps)	-21.00 ± 0.00	-20.30 ± 1.06	-20.56 ± 0.75	-20.47 ± 0.73
FGSM (1-Step)	-21.00 ± 0.00	-20.62 ± 0.75	$+20.75 \pm 0.43$	$+16.80 \pm 7.88$
ER2FGSM (20 steps)	-21.00 ± 0.00	-20.35 ± 0.8	-20.91 ± 0.29	-20.32 ± 0.95
ER2FGSM-v2 (20 steps)	-21.00 ± 0.00	-20.28 ± 0.9	-20.90 ± 0.36	-20.24 ± 0.91

Table 2: Comparison of Perturbation Methods in Average Reward (AR)

Perturbation Method	No Defense		Defense with SA-DQN	
	Non-targeted	Targeted	Non-targeted	Targeted
CW (1000 steps)	100%	97%	3%	0%
CW (20 steps)	100%	0%	3%	0%
PGD (20 steps)	100%	82%	99%	77%
DIFGSM (20 steps)	100%	73%	73%	54%
MIFGSM (20 steps)	100%	83%	75%	76%
FGSM (1 step)	85%	76%	28 %	19%
ER2FGSM (20 steps)	100%	83%	98%	77%
ER2FGSM-v2 (20 steps)	100%	84%	99%	83%

Table 3: Comparison of Perturbation Methods in Attack Success Rate (ASR)

Needs to be done after results, plots/graphs with respect to varying n, ϵ, and α

Perturbation method	AET under non-targeted attacks	AET under targeted attacks
CW (1000 steps)	716 ms $\pm$ 32 ms	963 ms $\pm$ 20 ms
MIFGSM (20 steps)	128 ms $\pm$ 8 ms	138 ms $\pm$ 8 ms
DIFGSM (20 steps)	103 ms $\pm$ 6 ms	135 ms $\pm$ 14 ms
PGD (20 steps)	94 ms $\pm$ 6 ms	117 ms $\pm$ 13 ms
CW (20 steps)	21 ms $\pm$ 2 ms	26 ms $\pm$ 2 ms
FGSM (1 step)	6 ms $\pm$ 2 ms	6.3 ms $\pm$ 0.7 ms
ER2FGSM (20 steps)	**126** ms $\pm$ 7 ms	**125** ms $\pm$ **7** ms
ER2FGSM-v2 (20 steps)	**82** ms $\pm$ 6 ms	**120** ms $\pm$ **7** ms

Table 4: Comparison Between Different Perturbation Methods Against Attack Execution Time Per Perturbation (AET) Under Targeted or Non-Targeted Attacks.

Conclusion & Future Work

This is the first of its kind research in DRL environment where we highlighted specifically on devising the need of robust and efficient state perturbation attacks by evaluating ER2FGSM with state of the art attacks in terms of attack success rate, attack execution time and robustness.

We obtained the following **key findings** from our study:

- ...
- ...
- ...

For the future work, we plan: (1) using various DRL algorithms and applications to further prove the effectiveness and efficiency of our proposed perturbation attacks; (2) adding more novel perturbation methods by using the concept of diversity to extend our attacks to blackbox setting (3) building novel efficient and defender-aware end goal based strategies to attack a defended DRL agent.

References

Alzantot, M.; Balaji, B.; and Srivastava, M. 2018. Did you hear that? adversarial examples against automatic speech recognition. *arXiv preprint arXiv:1801.00554*.

Amarjyoti, S. 2017. Deep reinforcement learning for robotic manipulation-the state of the art. *arXiv preprint arXiv:1701.08878*.

Basori, A. H.; and Malebary, S. J. 2020. Deep reinforcement learning for adaptive cyber defense and attacker's pattern identification. In *Advances in Cyber Security Analytics and Decision Systems*, 15–25. Springer.

Behzadan, V.; and Munir, A. 2017. Vulnerability of Deep Reinforcement Learning to Policy Induction Attacks. *CoRR*, abs/1701.04143.

Carlini, N.; and Wagner, D. 2017. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, 39–57. IEEE.

Dong, Y.; Liao, F.; Pang, T.; Su, H.; Zhu, J.; Hu, X.; and Li, J. 2018. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 9185–9193.

Ferdowsi, A.; Challita, U.; Saad, W.; and Mandayam, N. B. 2018. Robust deep reinforcement learning for security and safety in autonomous vehicle systems. In *The IEEE 21st International Conference on Intelligent Transportation Systems (ITSC)*, 307–312.

Fischer, M.; Mirman, M.; Stalder, S.; and Vechev, M. 2019. Online robustness training for deep reinforcement learning. *arXiv preprint arXiv:1911.00887*.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2014a. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2014b. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Huang, S. H.; Papernot, N.; Goodfellow, I. J.; Duan, Y.; and Abbeel, P. 2017. Adversarial Attacks on Neural Network Policies. *CoRR*, abs/1702.02284.

Kiran, B. R.; Sobh, I.; Talpaert, V.; Mannion, P.; Al Sallab, A. A.; Yogamani, S.; and Pérez, P. 2021. Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems*.

Kos, J.; and Song, D. 2017. Delving into adversarial attacks on deep policies. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net.

Kurakin, A.; Goodfellow, I.; and Bengio, S. 2016. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*.

Kurakin, A.; Goodfellow, I.; Bengio, S.; et al. 2016. Adversarial examples in the physical world.

Lin, Y.-C.; Hong, Z.-W.; Liao, Y.-H.; Shih, M.-L.; Liu, M.-Y.; and Sun, M. 2017. Tactics of Adversarial Attack on Deep Reinforcement Learning Agents. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, 3756–3762.

Luong, N. C.; Hoang, D. T.; Gong, S.; Niyato, D.; Wang, P.; Liang, Y.-C.; and Kim, D. I. 2019. Applications of deep reinforcement learning in communications and networking: A survey. *IEEE Comms. Surveys & Tutorials*, 21(4): 3133–3174.

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; and Riedmiller, M. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; et al. 2015. Human-level control through deep reinforcement learning. *nature*, 518(7540): 529–533.

Oikarinen, T.; Weng, T.-W.; and Daniel, L. 2020. Robust deep reinforcement learning through adversarial loss. *arXiv preprint arXiv:2008.01976*.

Pattanaik, A.; Tang, Z.; Liu, S.; Bommannan, G.; and Chowdhary, G. 2017. Robust deep reinforcement learning with adversarial attacks. *arXiv preprint arXiv:1712.03632*.

Sun, J.; Zhang, T.; Xie, X.; Ma, L.; Zheng, Y.; Chen, K.; and Liu, Y. 2020. Stealthy and Efficient Adversarial Attacks against Deep Reinforcement Learning. *Proc. the AAAI Conference on Artificial Intelligence*, 34(04): 5883–5891.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; and Fergus, R. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

Tramèr, F.; Kurakin, A.; Papernot, N.; Goodfellow, I.; Boneh, D.; and McDaniel, P. 2017. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*.

Xie, C.; Zhang, Z.; Zhou, Y.; Bai, S.; Wang, J.; Ren, Z.; and Yuille, A. L. 2019. Improving transferability of adversarial examples with input diversity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2730–2739.

Yoon, S.; Cho, J.-H.; Dixit, G.; and Chen, R. 2021a. Resource-Aware Intrusion Response Based on Deep Reinforcement Learning for Software-Defined Internet-of-Battle-Things. *Game Theory and Machine Learning for Cyber Security*.

Yoon, S.; Cho, J.-H.; Kim, D. S.; Moore, T. J.; Free-Nelson, F.; and Lim, H. 2021b. DESOLATER: Deep Reinforcement Learning-Based Resource Allocation and Moving Target Defense Deployment Framework. *IEEE Access*, 9: 70700–70714.

Yuan, X.; He, P.; Zhu, Q.; and Li, X. 2019. Adversarial Examples: Attacks and Defenses for Deep Learning. *IEEE Transactions on Neural Networks and Learning Systems*, 30(9): 2805–2824.

Zhang, H.; Chen, H.; Xiao, C.; Li, B.; Liu, M.; Boning, D.; and Hsieh, C.-J. 2020. Robust deep reinforcement learning against adversarial perturbations on state observations. *arXiv preprint arXiv:2003.08938*.